VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Externí instant messaging komunikační systém pro Virtlab

External instant messaging communication system for Virtlab

2012 **Michal Klimas**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Michal Klimas**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Externí instant messaging komunikační systém pro Virtlab
External Instant Messaging Communication System for Virtlab

Zásady pro vypracování:

Cílem práce je vytvářet sociální síť mezi uživateli systému Virtlab i v rámci různých lokalit. Hlavním přínosem by mělo být zefektivnění komunikace mezi spolupracujícími uživateli, dále možnost konzultovat řešení úloh s pokročilejšími uživateli a také zjednodušit technickou podporu.
1. Seznamte se s implementací systému Virtlab a s technologii XMPP/Jabber a různými implementacemi XMPP serveru. Na základě zjištěných informací uvažte, zda lze použít některý z dostupných veřejných XMPP serverů nebo bude lépe hostovat vlastní XMPP server.
2. Prozkoumejte dostupné existující webové klienty, případně navrhněte klienta vlastního a integrujte do webového rozhraní systému Virtlab.
3. Přihlášený uživatel by měl mít možnost vidět seznam právě přihlášených uživatelů v různých lokalitách a seznam jejich rezervovaných úloh (SOAP). Dále zajistěte automatickou registraci přihlášených uživatelů na Jabber server a ostatní přihlášené uživatele do jejich kontakt listu.
4. Celé řešení řádně otestujte a zdokumentujte.

Seznam doporučené odborné literatury:

GRAMES, Martin. Reimplementace řídící aplikace systému Virtlab s použitím moderních webových technologií. Ostrava, 2010. 47 s. Diplomová práce. VŠB-TU Ostrava, FEI.
Stránky projektu Virtlab [online]. 2006-06-28 [cit. 2011-03-01]. Virtuální laboratoř počítačových sítí. Dostupné z WWW: <http://www.virtlab.cz>.
Stránky o technologii XMPP/Jabber [online]. 2006-01-08 [cit. 2011-03-01]. XMPP/Jabber portál. Dostupné z WWW: <http://www.jabber.cz>.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Kateřina Bambušková**

Datum zadání: 18.11.2011
Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: 4.5.2012

.....

Poděkování

Velmi rád bych poděkoval Ing. Kateřině Bambuškové za odbornou pomoc a vedení při vytváření této bakalářské práce.

Abstrakt

Tato práce má za cíl vytvořit komunikační síť pro uživatele systému Virtlab za použití XMPP protokolu. Zabývá se výběrem a následným použitím XMPP serveru jako základ pro vytvoření komunikační sítě. Další část se zabývá výběrem dostupných webových XMPP klientů. Následně řeší implementaci vybraného webového XMPP klienta do webového rozhraní systému Virtlab.

Klíčová slova

XMPP, XMPP server, webový XMPP klient, Virtlab, Komunikační síť, webové rozhraní

Abstract

This thesis aims to create a communication network for users of Virtlab system using the XMPP protocol. It deals with the selection and subsequent using the XMPP server as the basis for the creation of a communication network. Another part deals with the selection of available Web XMPP clients. Next solves the implementation of selected Web XMPP clients to the web interface of Virtlab system.

Key words

XMPP, XMPP server, XMPP web klient, Virtlab, Communication network, Web interface

Seznam použitých zkratek

Virtlab – Virtuální laboratoř počítačových sítí.

XMPP – (Extensible Messaging and Presence Protocol) Rozšiřitelný protokol pro zasílání zpráv a zjištění stavu.

PHP – (Hypertext Preprocessor) Skriptovací programovací jazyk pro programování dynamických internetových stránek.

JID – (Jabber ID) Identifikace uživatele v XMPP síti.

IM – (Instant Messaging) Služba pro okamžitou komunikaci po síti.

XML – (Extensible Markup Language) Strukturovaný značkovací jazyk.

SSL/TLS – (Secure Socket Layer/Transport Layer Security) – kryptografické protokoly, poskytující zabezpečenou komunikaci po Internetu.

IETF – (Internet Engineering Task Force) – Skupina spravující standardy v síti Internet.

MUC – (Multi-user chat) – Víceuživatelský chat.

XSF – (XMPP Standards Foundation) - Nadace starající se o XMPP standardy.

XEP – (XMPP Extension Protocol) – Označení pro rozšíření XMPP protokolu.

SSH – (Secure shell) – Zabezpečený komunikační protokol počítačových sítí.

VPN – (Virtual private network) – Virtuální privátní síť.

JDK – (Java Development Kit) – Soubor nástrojů pro vývoj aplikací v prostředí Java.

JRE – (Java Run Enviroment) – Prostředí pro běh programů v Javě.

MySQL – (My Structured Query Language) – Strukturovaný dotazovací jazyk.

LDAP – (Lightweight Directory Access Protocol) – protokol pro ukládání a přístup k datům na adresářovém serveru.

HTTP – (Hypertext Transfer Protocol) – Protokol určený pro zasílání hypertextových dat.

BOSH (Bidirectional-streams Over Synchronous HTTP) – Technologie pro obousměrnou komunikaci přes HTTP protokol.

OBSAH:

1 Úvod

Základní koncepce Virtuální laboratoře počítačových sítí byla vytvořena v roce 2005 Ing. Pavlem Němcem pod vedením Ing. Petra Grygárka, Ph.D. Od této doby byl dále rozšiřován vývojovým týmem Virtlabu. V současné době se stále využívá pro praktickou výuku počítačových sítí na VŠB-TU Ostrava.

Hlavní důraz v této práci je kladen na komunikační protokol XMPP, který je využíván pro komunikaci mezi uživateli. V praxi jej využívají například společnosti Google a Facebook pro komunikaci ve svých službách v podobně Google Talk a Facebook chatu.

Cílem této práce je za pomocí XMPP protokolu vytvořit komunikační síť mezi uživateli systému Virtlab. Tato komunikační síť by měla zefektivnit spolupráci mezi uživateli, popřípadě mít možnost řešit daný problém s pokročilejšími uživateli (například učitelem).

V kapitole 2 popisuji základní implementací systému Virtlab a technologii XMPP. Kapitola 3 se zabývá výběrem a implementací XMPP Openfire serveru, který je nutným prvkem pro komunikaci mezi uživateli. Kapitola 4 představuje výběr webového klienta, za pomoci kterého jsou uživatelé schopni vzájemně komunikovat přes XMPP server. Jeho implementace do webového rozhraní systému Virtlab je popsána v kapitole 5.

2 Seznámení se systémem Virtlab a technologií XMPP/Jabber

2.1 Virtuální laboratoř počítačových sítí (Virtlab) [1]

2.1.1 O Virtlabu

Virtuální laboratoř počítačových sítí (dále jen Virtlab) je systém, který umožňuje vzdálené připojení k síťovým prvkům prostřednictvím Internetu a pracovat s nimi. Byl vytvořen za účelem procvičování a řešení praktických úloh studentů, a také ke zpřístupnění laboratorních síťových prvků, které byly v časech mimo výuku povětšinou nevyužity. Studenti si mohou pomocí webového rozhraní zarezervovat laboratorní prvky v určený čas a následně s nimi pracovat. Systém umožňuje spolupráci více lokalit mezi sebou, ty si vzájemně sdílejí síťové prvky přes Internet.

Systém Virtlab obsahuje informační systém, který zajišťuje správu uživatelů, rezervaci, zařízení, rezervaci úloh a spouštění úloh. Propojením tohoto informačního systému a použitím dynamického webového rozhraní, dostaneme prostředí pro plné ovládání systému Virtlab.

2.1.2 Komponenty/řídící server

Celý systém Virtlab se skládá z různých komponent a serverů, kde každá má svou roli a funkčnost. Tato práce se zaměřuje na komponentu, která se nazývá řídící server. Tento server se stará o celkové webové rozhraní. Zde se dostávají do styku s Virtlabem nejen běžní uživatelé, ale také správci úloh a administrátoři. Je zde implementována celková správa uživatelů, kteří jsou uloženi v databázi a pouze oni se mohou k webové aplikaci přihlásit. Obsahuje také databázi úkolů a rezervací, se kterými poté uživatel pracuje.

2.1.3 Shrnutí

Virtlab je celkově velmi rozsáhlý systém, ovšem v této práci se budeme zabývat jen částí spojenou se správou uživatelů a také v malé míře s rezervováním úloh.

2.2 Technologie XMPP/Jabber [2][3][4]

2.2.1 Historie

XMPP je zkratka pro Extensible Messaging and Presence Protocol, neboli česky „rozšiřitelný protokol pro zasílání zpráv a zjištění stavu“. Je to otevřený protokol pro komunikaci ve skoro-reálném čase, anglicky označován jako Instant Messaging (dále jen IM). Tento protokol vychází z otevřeného (open-source) projektu Jabber, který založil v roce 1998 Jeremie Miller. V roce 2004 byl na základě použitého protokolu vytvořen standart XMPP. V dnešní době je Jabber velice obecný název, lidé jím

označují ledasco, od protokolu přes komunikační sít až po různá XMPP rozšíření. JABBER je ochranná známka společnosti Jabber Inc. Proto mnoho poskytovatelů XMPP řešení slovo Jabber nepoužívá. Je třeba zdůraznit, že XMPP je schválen jako standart pro IM organizací IETF[1].

XMPP Standards Foundation (zkráceně XSF) je nezávislá, nezisková organizace, která se stará o standardizaci, vývoj a správu XMPP Extension Protokol (XEP), což je rozšíření o protokoly, které jsou nad rámec XMPP standardu. Tato organizace se dále stará o podporu a rozvoj Jabber/XMPP vývojářské komunity.

2.2.2 Možnosti XMPP

Celá implementace XMPP protokolu a mnoho klientských aplikací jsou distribuovány jako Open Source, to znamená, že uživatel může za jistých podmínek zdrojový kód používat či upravovat a to zcela legálně. Díky tomu existuje spousta serverů a klientů běžících na nejrůznějších operačních systémech a platformách, uživatel tak není vázán a má možnost výběru.

Oproti ostatním IM není XMPP centralizovaný do jediného místa, ale zastupuje celou síť serverů. Pokud vypadne nějaký server, zasáhne to jen malou část uživatelů. Kdokoli může provozovat vlastní XMPP server, i na vlastní privátní síti, kde budeme mít jen své uživatele a je na něm zdali se připojí ke zbytku světa s ostatními servery.

Další z mnoha výhod protokolu XMPP je bohatá nabídka rozšíření, které nám zpřístupňují například víceuživatelskou konferenci, transporty[2], zjišťování předpovědi počasí, televizního programu, čtení RSS, příjem e-mailů a spousta dalších. XMPP byl navržen tak, aby byl snadno rozšířitelný, a tak každý může přispět o nové funkce.

XMPP má i své nevýhody, jednou z nich je funkce Heartbeat, která není v XMPP podporována. Jedná se o funkci, která zajišťuje pravidelnou kontrolu přihlášených uživatelů. V XMPP každý klient před odpojením posílá na server zprávu, že je uživatel odhlášen. Tím dá ostatním vědět, že je již nedostupný. Může však nastat situace, kdy daný klient nestihne odeslat zprávu o odhlášení (nečekaný pád klienta, přerušení síťové komunikace), pro ostatní je ale stále viditelný jako přihlášený až do doby, kdy mu při odeslání zprávy přijde nazpět chybová hláška.

2.2.3 Způsob komunikace

Každý uživatel daného serveru má své jedinečné uživatelské jméno, označuje se Jabber ID (zkráceně také JID). Pomocí tohoto jména se uživatel přihlašuje ke svému XMPP účtu. Jsou podobná e-mailovým adresám a obvykle se skládají z tvaru uživatel@doména/zdroj. Na svůj uživatelský účet

[1] IETF (Internet Engineering Task Force) – Skupina spravující standardy v síti Internet.
[2] Transporty – jsou služby XMPP serveru, které umožňují komunikovat s jinými IM protokoly.

se lze přihlásit z více míst v jeden okamžik, proto se za lomítkem udává zdroj. Povětšinou jméno používaného klienta, odkud dochází k připojení (např.: vojtapulec@jabbim.cz/Miranda).

XMPP síť používá architekturu klient-server, kdy klienti zpravidla nekomunikují přímo, ale přes server. Následující příklad ukazuje, co se děje při zaslání zprávy od jednoho uživatele ke druhému:

- Máme uživatele Marka, který je zaregistrovaný na svém vlastním serveru s doménovým jménem „kov.cz“. Druhý uživatel Toník, zaregistrovaný na serveru „hutnik.net“. Marek pošle zprávu Toníkovi, následují tyto akce.
- XMPP klient Marka zašle zprávu serveru „kov.cz“.
- Server „kov.cz“ se podívá, zda má přístup na server „hutnik.net“.
    - Pokud je server „hutnik.net“ blokován, server s ním není propojen, zpráva je smazána a nazpět je zaslána chybová zpráva.
    - V opačném případě se otevře spojení se serverem a předá mu danou zprávu.
- Server „hutnik.net" doručí zprávu XMPP klientovi uživatele Toník.
    - Pokud není Toník právě připojen, uloží se zpráva na server a doručí se při nejbližší příležitosti.

Komunikace pomocí protokolu XMPP probíhá výměnou XML[3] dat. Pro přenos zpráv a jinou komunikaci lze také použít šifrování SSL/TLS[4], popřípadě jiné zabezpečení, které server nabízí. Zprávy jsou kódovány pomocí sady Unicode[5], obsahující všechny znaky světa.

[3] XML (Extensible Markup Language) – obecný značkovací jazyk.

[4] SSL/TLS (Secure Socket Layer/Transport Layer Security) – kryptografické protokoly, poskytující zabezpečenou komunikaci po Internetu.

[5] Unicode – Standart pro kódování znaků, obsahuje všechny možné znaky světa.

3 Výběr a implementace XMPP serveru

3.1 Vlastní XMPP server

3.1.1 Důvody hostování vlastního serveru

Mým úkolem bylo rozhodnout, zda využít veřejně přístupné XMPP servery na Internetu, nebo zda bude lepší hostovat vlastní server. Rozhodl jsem se pro možnost hostování vlastního serveru a to hned z několika důvodů.

Jeden z hlavních důvodů je, mít plnou kontrolu nad správou svých uživatelů, nastavení MUC[6] místností a ostatních administrátorských práv. Většina veřejných XMPP serverů má vlastní podmínky pro užívání jejich služeb a povětšinou nedovolují nastavení práv uživatelům podle našich představ. Tyto servery také mají spoustu svých uživatelů, kteří by nás mohli narušovat, či jinak ovlivňovat komunikaci mezi našimi uživateli.

Dalším důvodem hostování vlastního XMPP serveru je, že máme k dispozici vhodné zařízení, které nám to umožňuje. Každá lokalita systému Virtlab má svůj server, na kterém běží jeho komponenty potřebné k provozu. Na každou tuto lokalitu máme možnost nainstalovat a spravovat XMPP server, tím můžeme vytvořit vlastní komunikační síť na vlastní doméně. Tato možnost instalace a celkový provoz nám může dát nové a zajímavé zkušenosti o této technologii.

3.1.2 Výhody a nevýhody hostování vlastního serveru

Hlavními výhodami jsou:

- Celková administrace serveru je plně pod naší kontrolou.
- Správa nad uživateli, skupinami, MUC.
- Možnost nastavit propojení s jinými servery mezi s sebou i se servery na Internetu.
- Rozdělení na každou lokalitu jeden XMPP server, pád jednoho serveru neovlivní ostatní uživatele na jiných lokalitách.

Mezi nevýhody patří:

- Jedna z nevýhod oproti veřejným serverům je někdy složitější prvopočáteční instalace a uvedení do provozuschopného stavu. To se ale podstatně liší od různých druhů XMPP serverů.
- Následná správa celého serveru a udržování v bezproblémovém stavu, popřípadě řešení různých chyb a závad. Což vyžaduje náš čas a úsilí.

[6] MUC (Multi-user Chat) – Víceuživatelská komunikace v jednom okně.

3.2 Výběr XMPP serveru

Protože máme na výběr z několika desítek produktů řešící XMPP server, tak jsem si musel stanovit kritéria. Podle nich jsem se rozhodoval, jaký server bude nejoptimálnější pro naše potřeby. Shrnul jsem je do několika bodů:

- **Licence** – I když je XMPP protokol označen jako Open-source projekt, některé druhy serverů jsou licencovány jako komerční. My budeme vybírat z produktů licencovaných jako svobodný software.
- **Operační systém** – Celý systém Virtlab běží v operačním systému Linux, my budeme tento server používat i pro XMPP server, proto musíme vybírat se serverů běžící pod tímto operačním systémem.
- **Pro jaké potřeby to chceme využit** – Musíme si stanovit, jaké služby by měl server podporovat. Pro naše potřeby budeme využívat jen základní služby a to klasickou komunikaci mezi dvěma uživateli a více uživatelskou komunikaci.
- **Zatíženost** – Tady je nutné brát ohled na dvě stránky věci. První si stanovíme, kolik uživatelů budeme spravovat a jaký počet jich bude přibližně aktivních ve špičce. Stanovil jsem počet v řádech desítek uživatelů na jednu lokalitu. Podle toho vybereme server, který je zaměřený buďto na velkou nebo na malou správu uživatelů. V druhé stránce věci se musíme zaměřit na to, jak bude daný server svými procesy zatěžovat náš Linuxový server a jestli nebude ohrožovat ostatní procesy svým chováním.
- **Dokumentace a snadná implementace** – Při vybírání jsem bral ohled také na složitost instalace a správu daného serveru. Především přehledný popis serveru a také dobrý návod instalace hrál velkou roli při rozhodování.

Při vybírání serveru jsem vycházel ze seznamu, uveřejněném organizací XSF na jejich internetových stránkách. Seznam se skládá ze jména softwaru serveru, platformě, licence a z odkazu, který odkazuje na případné stránky projektu [3]. To mi hodně usnadnilo vyhledávání a vybírání výsledného serveru. Rozhodoval jsem se nakonec ze čtyř serverů, mezi ně patří:

- **Ejabberd** – Server je zaměřený především pro velké množství uživatelů. Pro naše potřeby zahrnuje až příliš velké množství potřebných komponent, které potřebuje pro svůj běh. Z toho vyplývá i celková náročnost instalace. Pro velké projekty s velkým množstvím aktivních uživatelů (řádově tisíce), bych zvolil zcela určitě tento server. Tady musím upozornit na vhodně a vystižně popsanou dokumentaci. [5]
- **Jabberd14** – Bohužel tento server jsem zavrhl kvůli celkově špatné dokumentaci. [6]
- **Prosody** – Jednoduchý server pro menší množství uživatelů. Jeho provoz nijak zvlášť nezatěžuje systém, na kterém běží. Dokumentace i instalace je výstižně a pochopitelně

napsána. Prvotní nastavení, uvedení do provozu a následná správa serveru je řešena přes příkazovou řádku. Je vhodným kandidátem pro realizaci a následnou implementaci. [7]

- **Openfire** – Velmi rozšířený XMPP server, vyniká svou jednoduchou instalací a přehlednou dokumentací. Konfigurace a následný provoz je prováděn v přehledném webovém rozhraní, které má navíc podporu češtiny i několika dalších jazyků. Je určen pro menší i velký počet uživatelů, není tím nějak zvlášť omezen. Nakonec jsem se rozhodl pro implementaci tohoto serveru. [8]

3.3 Openfire XMPP server

Openfire XMPP server je vyvíjený společností Jive Software a je napsán v programovacím jazyce Java. Server je dostupný pod licencí Open Source Apache, to nám umožní používat software pro jakékoliv účely. Jak jsem již zmiňoval, Openfire má velmi přehledné webové rozhraní pro správu a jednoduchou instalaci. Běží na operačních systémech Linux, Windows, Mac OS X a dalších. [8]

3.3.1 Přidělená virtuální lokalita

Pro mé účely testování a implementování celé úlohy, mi byla přidělena jedna lokalita Virtuálního Virtlabu. Virtuální Virtlab je prostředí, které svými schopnostmi kopíruje reálnou virtuální laboratoř. Jeho základ tvoří jediný server, na kterém je provozováno množství rozšíření a modifikací simulující celkové vybavení lokality, jakožto i pro naši práci důležitý řídící server. Toto virtuální prostředí je určeno jen pro účely testování dalšího vývoje reálného distribuovaného Virtlabu.

Virtuální lokality jsou přístupné pouze z IP adresy školy pomocí protokolu SSH[7]. Pro přístup mimo síť školy je zapotřebí použít služby VPN[8]. My budeme pracovat na přidělené lokalitě s doménovým jménem `most.dvirtlab.net`, na kterém budu pracovat s administrátorským oprávněním (root).

3.3.2 Implementace

Implementace Openfire serveru je poměrně jednoduchá. Protože je server napsaný v Javě, je potřeba pro jeho chod mít nainstalovaný JDK[9] nebo JRE[10] 1.5.0 (nebo novější). Kontrolu verze Javy můžeme provést následujícím příkazem:

```
#java -version
```

[7] SSH (Secure shell) – Zabezpečený komunikační protokol počítačových sítí.
[8] VPN (Virtual priváte network) – Zabezpečené propojení do privátní sítě přes internet.
[9] JDK (Java Development Kit) – Soubor nástrojů pro vývoj aplikací v prostředí Java.
[10] JRE (Java Run Enviroment) – Prostředí pro běh programů v Javě.

V případě potřeby aktualizujeme ze stránek http://java.sun.com. Následně si stáhneme balíček poslední stabilní verze Openfire a rozbalíme do souborové složky `/opt`[11].

```
#tar -xzvf openfire_3_7_1.tar.gz
#mv openfire /opt
```

Openfire používá svou vlastní interní databázi HSQLDB[12], i když není příliš doporučována, protože při vyšším počtu uživatelů narůstá i zatížení serveru. Openfire nám dává možnost použít různých externích databází, kdy při použití výrazně snížíme zatížení celého serveru. My tuto možnost využijeme a použijeme MySQL[13] databázi, která je již používána řídícím serverem a tudíž máme většinu věcí předpřipravených.

Pro použití externí databáze MySQL pro Openfire je potřeba verze 4.1.18 a vyšší. Na naší lokalitě běží databáze MySQL 5.1.49 tuto podmínku splňujeme. Nyní potřebujeme vytvořit databázi pro Openfire tabulky. Abych se moc nelišil od syntaxe již vytvořených databází pro řídící server (`dvldb-reser` databáze řešící rezervaci úloh a `dvldb-www` databáze řešící webové rozhraní a správu uživatelů), zvolil jsem jméno tabulky `dvldb-xmpp`. Kde zkratka dvldb znamená distribuovaná virtuální laboratoř databáze a xmpp znázorňuje náležitost pro daný protokol XMPP. Pro vytvoření dané databáze použijeme příkaz:

```
#mysqladmin create dvldb-xmpp
```

Po vytvoření databáze nás čeká ještě importace souboru schématu tabulek, se kterýma dále Openfire pracuje. To můžeme udělat dvojím způsobem. Já jsem volil cestu přes phpMyAdmin, což je nástroj pro správu MySQL databáze. Do naší vytvořené databáze jsem pak importoval soubor `openfire_mysql.sql`, který nalezneme v adresáři nainstalovaného Openfire serveru `/opt/openfire/resources/database` . Druhá možnost je soubor do databáze importovat příkazem:

```
#cat openfire_mysql.sql | dvldb-xmpp
```

Pro obě tyto operace bych doporučil použít webový nástroj phpMyAdmin, je přehlednější, vše můžeme snadno zkontrolovat a máme jistotu, že vše proběhlo v pořádku. Nakonec musíme vytvořit uživatele, kterému přidělíme plná práva administrace dané databáze. Tohoto uživatele poté při konfiguraci přiřadíme Openfire serveru, který pod tímto účtem bude spravovat danou databázi. Teď již máme vše připravené na spuštění Openfire serveru a následnou konfiguraci.

[11] /opt – základní adresář pro umístění volitelných aplikací.
[12] HSQLDB – je relační databázový systém napsaný v Javě.
[13] MySQL – multiplatformní relační databázový systém.

Openfire server se spouští z příkazového řádku příkazem:

```
#/opt/openfire/bin/openfire start
```

Zdali Openfire server běží, můžeme zjistit příkazem:

```
#/opt/openfire/bin/openfire status
```

Po prvním spuštění server stále ještě nepoběží, musíme ho nejprve nakonfigurovat. K tomu nám slouží administrátorské webové rozhraní, ke kterému se připojujeme pomocí internetového prohlížeče. Standardně běží na portu 9090, pokud se připojujeme ze stejného stroje, kde máme nainstalovaný Openfire, bude URL adresa následující (http://127.0.0.1:9090/). Pokud se připojujeme z internetu, použijeme IP adresu či doménové jméno dané virtuální lokality, pro naši virtuální lokalitu bude URL adresa vypadat takto (http://most.dvirtlab.net:9090/). Pro zabezpečené připojení lze použit port 9091.

Po příchodu na webové administrační rozhraní se nám zobrazí úvodní konfigurace, kde nastavíme základní parametry Openfire serveru a až poté se dostaneme na pokročilejší správu serveru. Prvotní konfigurace se skládá z pěti kroků:

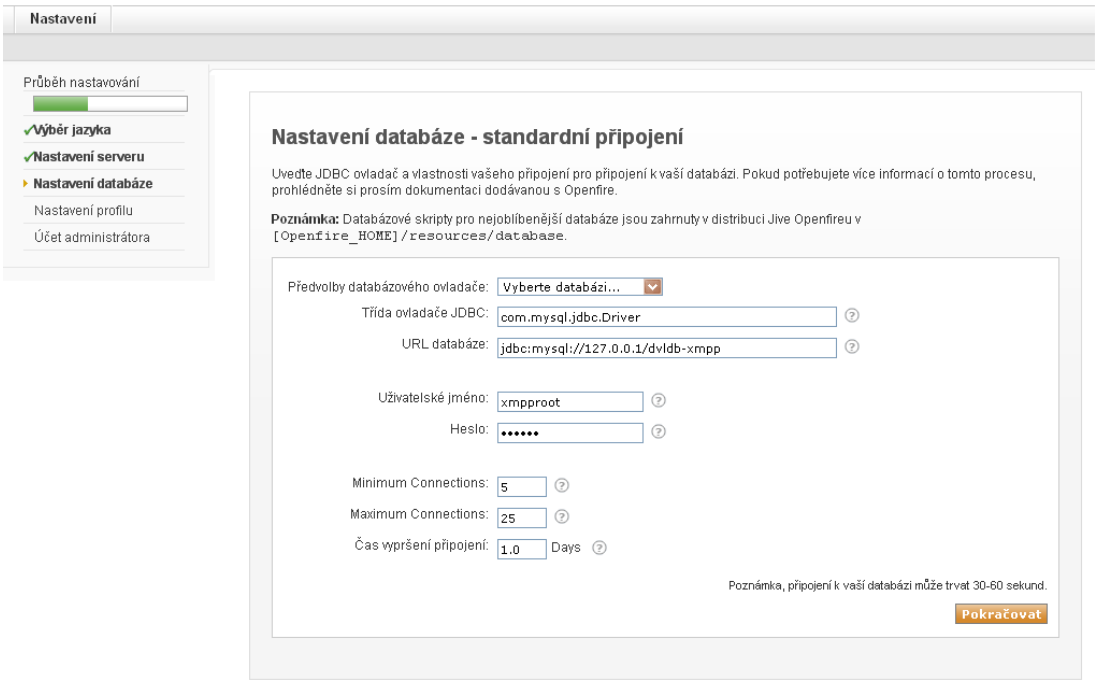
1) Prvním krokem je výběr jazyka. Tento jazyk nás bude doprovázet celým webovým administrátorským rozhraním.
2) V Druhém kroku nastavujeme název domény, pod kterým bude server vystupovat. Ovlivňuje všechna JID na serveru, je to hodnota uvádějící se za zavináčem (jmeno@název.domény.net). Je doporučeno nastavit název domény podle doménového jména lokality, na kterém server poběží. Název domény jsem stanovil následovně:
   - `most.dvirtlab.net`

   Dále nastavujeme porty pro nešifrovaný (defaultně: 9090) a šifrovaný (defaultně: 9091) přístup k administrátorskému webovému rozhraní. Tyto hodnoty lze změnit, já však ponechal hodnoty defaultní.
3) Třetí krok nám dává na výběr, zda budeme chtít použít vestavěnou interní databázi, nebo použijeme externí databázi. Zvolil jsem použití externí databáze. Tuto databázi jsem si předem připravil a následně propojit s Openfire serverem vyplněním formuláře (viz. Obrázek 3.1). Ten obsahuje:
   - Výběr databázového ovladače: `MySQL`
   - Jméno ovladače JDBC: `com.mysql.jdbc.Driver`
   - URL pro připojení k naší databázi: `jdbc:mysql://127.0.0.1/dvldb-xmpp`

- Dále jméno a heslo uživatele, kterým se bude Openfire server připojovat k dané databázi. Uživatele s danými právy jsme si připravili při vytváření MySQL databáze.
- Další prvky jsme již nechali na defaultních hodnotách. Jedná se o nastavení počtu a doby databázových připojení, které má zásobník připojení udržovat.

4) Ve čtvrtém kroku volíme systém uživatelů a skupin, který má Openfire používat. Máme na výběr ukládání uživatelů a skupin v databázi používanou Openfire, nebo možnost integrovat s adresářovým serverem OpenLDAP[14] či Active Directory[15] za použití protokolu LDAP[16]. My použijeme volbu propojení s databází Openfire, kde budeme spravovat své vlastní uživatele a skupiny.
5) V posledním kroku nastavíme administrátorský účet. Defaultní jméno je ‚admin', zadáme e-mailovou adresu administrátora a heslo, pod kterým se budeme pro příští správu k administrátorskému rozhraní přihlašovat.

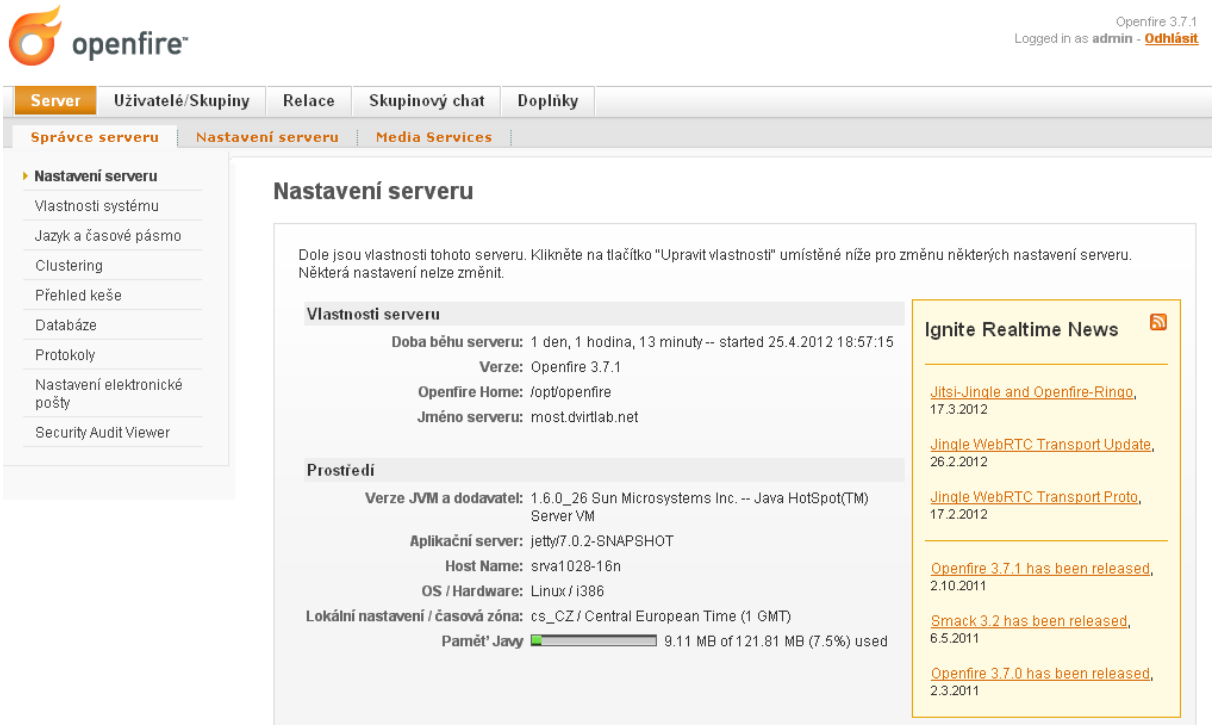
Po dokončení základní konfigurace jsme odkázání na hlavní administrátorské webové rozhraní. Od této doby jsme schopni Openfire XMPP server plně používat. Znovu spuštění úvodní konfigurace provedeme upravením řádku `<setup>true</setup>` (na `false`) v souboru openfire.xml v instalačním Openfire adresáři.



Obrázek 3.1 Administrační rozhraní – Prvotní nastavení (ukázka)

[14] OpenLDAP – Open source implementace LDAPu.
[15] Active Directory – implementace adresářových služeb LDAP.
[16] LDAP (Lightweight Directory Access Protocol) – protokol pro ukládání a přístup k datům na adresářovém serveru.

Základní konfiguraci pro chod serveru máme hotovou. Po přihlášení do administračního webového rozhraní máme možnost detailněji konfigurovat jeho vlastnosti. Nabízí se nám také správa uživatelů a skupin, kde je můžeme jednotlivě přidávat či jinak upravovat. Můžeme také pomocí tohoto rozhraní sledovat relace uživatelů, kteří jsou právě přihlášeni. Tato možnost se nabízí i pro právě připojené servery z jiných lokalit. Mezi další vlastnosti webového rozhraní patří správa místností víceuživatelského chatu a jeho nastavení. V Poslední části administračního rozhraní se setkáme se správou doplňků. V této části máme možnost stáhnout a nainstalovat různé doplňky, které Openfire server rozšiřují o novou funkčnost (ukázka administračního webového rozhraní Obrázek 3.2).



Obrázek 3.2 Administrační rozhraní – správa Openfire serveru

Popis a nastavení celého administračního webového rozhraní Openfire není nutné příliš rozepisovat. Popíšeme si jen věci, které bylo třeba pozměnit od výchozích pro naše požadavky.

Protože chceme, aby se na náš server přihlašovali jen registrovaní uživatelé, je potřeba zakázat přihlašování anonymních uživatelů (Server > Nastavení serveru > Registrace a přihlášení). U kolonky anonymní přihlášení, zvolíme možnost zakázat a nastavení uložíme.

Openfire server poskytuje možnost uložit a přeposlat zprávy uživatelům, kteří nejsou právě dostupní. Rozhodl jsem se tuto možnost vypnout. My chceme používat komunikaci jen pro zasílání zpráv v reálném čase. Pro zasílání zpráv a následné pozdější vyzvednutí slouží systém zpráv řešený

řídícím serverem Virtlabu. V nastavení *Server > Nastavení serveru > Offline zprávy* jsem zvolil možnost „vrátit“. To znamená, že odeslaná zpráva offline uživateli se neuloží, ale odešle jí nazpět. Poté už záleží na používaném klientovi, zda vypíše chybovou hlášku, nebo obsah dané zprávy.

Nastavení propojení Openfire serverů mezi sebou. Při testování na naši jedné virtuální lokalitě není nutné nastavovat propojení serveru mezi sebou. Poté až budeme aplikovat Openfire server na distribuované lokality Virtlabu, musíme nastavit mezi nimi propojení, aby mohli spolu komunikovat a tím vytvořit jednu společnou síť. V nastavení *Server > Nastavení serveru > Server / server* povolíme danou službu a nastavíme port, na který se vzdálené servery budou připojovat (výchozí hodnota portu: 5269). Dále máme možnost nastavit připojení jakéhokoliv serveru, nebo povolit připojení jen určitým serverům. My povolíme připojení námi definovanému seznamu serverů, do kterého pak přidáváme jména domén serverů a jejich portů, se kterými se chceme propojit.

Poslední věc v nastavení serveru, kterou potřebujeme upravit je HTTP[17] připojení, které umožňuje klientů používat HTTP protokol pro připojení k Openfire. O to se stará BOSH[18] server integrovaný v Openfire. My tuto funkci vypneme, protože budeme používat externí BOSH server, z důvodu, který rozebereme v části věnované implementací klienta. Zvolíme zakázat připojení se serverem pomocí HTTP připojení *Server > Nastavení serveru > HTTP Binding*.

Abychom jednotlivé uživatele rozlišili podle různých lokalit, rozhodl jsem se vytvořit na každém serveru skupinu s názvem dané lokality. Do této skupiny budou patřit všichni uživatelé lokality. Pro naši přidělenou virtuální lokalitu to bude skupina s názvem Most. Vytvoříme si skupinu *Uživatelé/skupiny > Skupiny > Vytvořit novou skupinu* a nastavíme, s kým jí chceme sdílet. Povolením sdílení skupiny v seznamu kontaktů přidělíme seznam uživatelů dané skupiny všem uživatelům do jejich Rosteru[19]. Dále musíme nastavit sdílení skupiny se skupinami z ostatních serverů. Tím se vyřeší, že všichni uživatelé celé komunikační sítě se navzájem uvidí.

Openfire server disponuje různými doplňky, které lze nainstalovat a dále využívat. My jsme jeden takový doplněk použili. Jedná se o rozšíření pro registraci uživatelů, které nabízí automatické akce při registraci nového uživatele. Tento doplněk najdeme v seznamu dostupných doplňků *Doplňky > Dostupné doplňky* s názvem Registration. Na tyto doplňky však není jazyková podpora, tudíž jsou v angličtině. Po nainstalování následuje nastavení prvků, které chceme využívat *Uživatelé/skupiny > Uživatelé > Registration properties*. Jediné čeho chceme využít je automatické zařazení do uživatelské skupiny, toho dosáhneme označením možnosti „Enable automatically adding of new users to a group“,

[17] HTTP (Hypertext Transfer Protokol) – Internetový protokol určený pro zasílání hypertextových dat.

[18] BOSH (Bidirectional-streams Over Synchronous HTTP) – Technologie pro obousměrnou komunikaci přes HTTP protokol.

[19] Roster – Je seznam kontaktů daného uživatele. Tento seznam je uložen vždy na serveru.

ostatní možnosti necháme neoznačené. Je potřeba nastavit do jaké skupiny chceme nově registrované uživatele přiřadit. Zvolíme již vytvořenou skupinu s názvem virtuální lokality (Most).

Pro komunikaci mezi více uživateli najednou je třeba vytvořit a nastavit službu skupinového chatu (Skupinový chat> Nastavení skupinového chatu > Create New Service). Službu pojmenujeme názvem `conference`. název nám udává část v JID vytvořených MUC místností (názevchatu@conference.most.dvirtlab.net). Celkové nastavení této služby udává, jaké vlastnosti budou mít všechny vytvářené MUC místnosti. Nastavíme historii skupinového chatu pro čerstvě příchozí uživatele na 10 řádků (při vstupu do místnosti se uživateli zobrazí posledních 10 zpráv chatu). Dále nastavíme možnost vytvoření místnosti jakémukoli uživateli. Následuje vyplnění formuláře pro výchozí nastavení MUC místností.

Nyní je Openfire server nastaven a je připraven pro všechny naše potřeby.

4 Stanovení požadavků a výběr webového klienta

XMPP klient je jakákoliv aplikace, či software, který nám poskytuje komunikaci s XMPP serverem a tím i s ostatními uživateli. Existuje mnoho klientů, kteří jsou zaměřeni přímo pro XMPP komunikaci. Najdou se i takzvaně multiprotokoloví klienti, kteří podporují několik protokolů najednou. Tito klienti běží pod různými systémy či platformami.

Protože vytvoření vlastního XMPP klienta je poněkud náročnější než jsem předpokládal, dal jsem možnost výběru z již vytvořených klientů, které jsou dostupné na Internetu. Při vybírání XMPP klienta jsem vycházel ze seznamu uveřejněném organizací XSF na jejich internetových stránkách. Stejně jako u výběru XMPP serveru, i zde je důležité, abychom si stanovili základní požadavky a předpoklady pro výběr XMPP klienta.

4.1 Stanovení požadavků

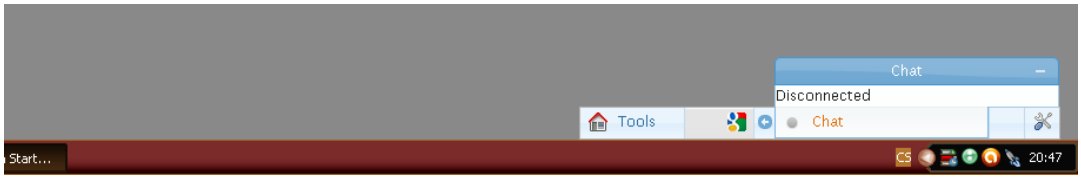
- **Operační systém** – Klienty bych rozdělil do tří základních kategorií, podle kterých rozlišujeme, pod jakou platformou pracují. Do první kategorie bych zařadil desktopové klienty, pracující pod operačními systémy jako je Windows, Linux, MAC OS X. Do druhé kategorie klienty, kteří běží v mobilních zařízeních, pod různými systémy jako je Android, iOS, JavaME a jiné. Do třetí kategorie bych zařadil ten zbytek, který se nedá zařadit do předchozích kategorií. Tím jsou hrací konzole anebo pro nás nejdůležitější klienti pracující ve webovém rozhraní. Pro naši práci budeme potřebovat klienta, který bude umět pracovat ve webovém rozhraní.
- **Jazyk** – Při výběru webové klienta jsem bral ohled na to, jakým programovacím jazykem je napsán a pod jakým Frameworkem běží. Vybíral jsem především klienty psané v PHP jazyce, Javaskriptu, Javě a podobných. Z důvodu toho, abychom se vyhnuli zbytečné instalaci doplňků do Internetových prohlížečů na straně uživatele.
- **Licence** – Jako u výběru serveru i tady jsem vybíral z produktů licencovaných jako svobodný software, bezplatně užívaný.
- **Implementace** – Důležitou roli při výběru hraje struktura a rozvržení použitého klienta. Naše požadavky jsou, aby byl klient zaprvé ve stejném okně daného prohlížeče a zadruhé, aby nijak zvlášť nezasahoval do struktury uživatelského rozhraní Virtlabu. Těmito podmínkami jsem chtěl dosáhnout, aby práce ve Virtlabu a zároveň komunikace s ostatními nebyla navzájem omezující.
- **Využití služby**- Poslední věcí, na kterou jsem bral ohled při výběru klienta, je, jaké možnosti mi může poskytnout. Pro naše účely jsou důležité tři věci. Klient by měl umět komunikovat s více uživateli najednou, tím mám na mysli komunikaci mezi dvěma

uživateli, ale ve více komunikačních oknech. Dále by měl umět víceuživatelský chat. Možnost zobrazit Roster uživatele a jeho seznam dostupných víceuživatelských chat místností, do kterých má možnost se přihlásit.

4.2 Výběr webového klienta

Výběr webového XMPP klienta byl poněkud obtížnější, než výběr XMPP serveru. Při výběru se mi naskytlo kolem 15 webových klientů licencovaných jako volný software, jak už od známých společností (Trillian, Jappix, SparkWeb), tak i od méně známých (iJab, Prodromus, Slimster). Bohužel velká většina těchto klientů funguje v samostatném okně a nelze je přímo implementovat do webového rozhraní Virtlabu. Při postupném zkoumání jsem narazil na dva reálně použitelné klienty, kteří ve většině případů vyhovovali mým požadavků. Výběr těchto klientů jsem si seřadil podle toho, jak nejlépe splňují tyto požadavky a jak je postupně budu testovat a implementovat do webového rozhraní Virtlabu. Jedná se o tyto klienty:

- **iJab** – U tohoto klienta mě zaujalo jeho webové rozhraní ve stylu Facebook či Gmail chat baru, které je umístěno ve spodní části internetového prohlížeče (Obrázek 4.1). Tento klient zahrnuje krom klasické komunikace mezi dvěma uživateli i víceuživatelský chat a jeho seznam místností, což náramně vyhovuje naším potřebám. Bohužel jedna věc, která mě nemile překvapila, je celkově špatná dokumentace a informovanost o tomto klientu. Návod na instalaci je sice v nějaké základní podobě k dispozici, ale to je tak asi vše. Většina nebo spíše všechny odkazy odkazující na blogy zabývající se tímto klientem jsou již nefunkční. Klient je napsaný pomocí Javaskriptu a je licencovaný jako GPL2[20]. [9]

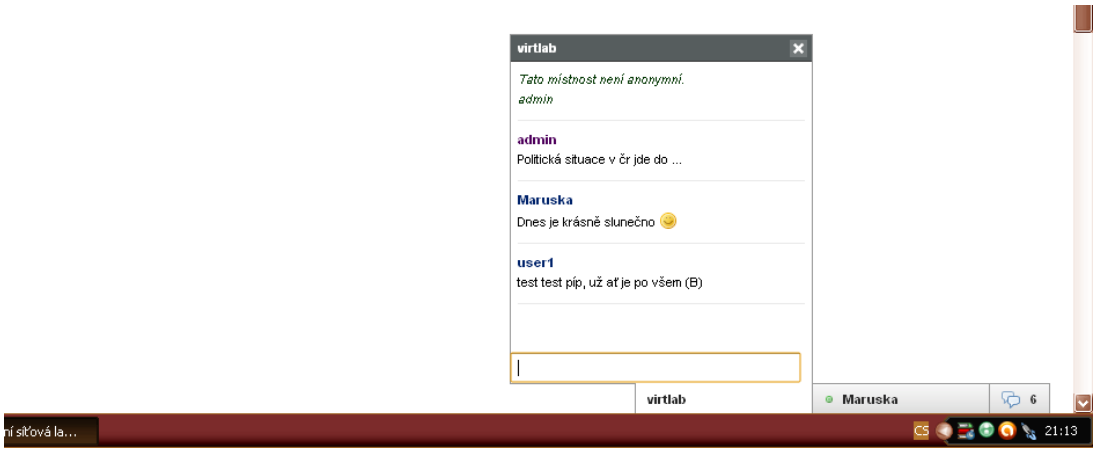


Obrázek 4.1 iJab komunikační rozhraní

- **Jappix** – Jappix je volná sociální sít, založena na protokolu XMPP. Poskytuje hlavního klienta, kterým se uživatel připojí do vlastní XMPP sítě a komunikuje dále s uživateli. Jappix nabízí i rozšíření v podobě klienta pro mobilní zařízení nebo mini klienta pro vlastní webové stránky. Já jsem se zaměřil na onoho mini klienta označovaného jako Jappix-Mini. Rozhraní je řešeno podobně jako u iJabu v podobě spodní lišty umístěné

[20] GPL2 (General Public Licence v. 2) – Všeobecná veřejná licence pro svobodný software 2. verze.

ve spodní části internetového prohlížeče (Obrázek 4.2). Podporuje komunikaci mezi dvěma uživateli i víceuživatelskou komunikaci. Bohužel obsahuje jen *Roster* uživatele, seznam místností už nepodporuje. Připojení k víceuživatelskému chatu je řešeno pomocí odkazu, kam musíme napsat jméno místnosti, poté jsme připojeni a můžeme komunikovat s uživateli. Celý Jappix je napsán v PHP a Javascriptu. Jappix je svobodný software a je šířen pod licencí AGPL[21]. Výborná dokumentace i uživatelská podpora je k dispozici. Instalace a nastavení je snadné pomocí webového rozhraní. [10]



Obrázek 4.2 Jappix-Mini komunikační rozhraní

Rozhodl jsem se pro vyzkoušení a implementování XMPP klienta iJab. Prostudoval jsem si všechny dostupné materiály, které byly k dispozici na stránkách projektu [9]. Bohužel jak jsem již zmiňoval, tento projekt postrádá řádnou dokumentaci, či technickou podporu. Nalezl jsem jen pár krátkých návodů na nastavení a implementaci klienta, ale i to mi stačilo a pustil jsem se do instalace. Stáhl jsem nejnovější dostupný balíček a ten rozbalil. Podle návodu jsem nakonfiguroval případný konfigurační soubor, který se měl připojit na můj Openfire server. Další nastavení směřovalo k nastavení našeho webového serveru Apache[22]. Po dokončení všelijakých nastavení popisovaných v návodu a následném spuštění se mi onen klient nepodařilo spustit korektně. Webové rozhraní klienta běželo správně, ale bohužel se klient stále nemohl připojit na náš Openfire server. Snažil jsem se najít chybu v nastavení ale neúspěšně. Při hledání možných chyb na internetových stránkách projektu jsem také na nic pozitivního nenarazil. Možná to bylo z důvodu, že většina odkazů, na které jsem narazil, byla nepoužitelná a nefunkční. Uživatelské fórum bylo také nedostupné a nikde jinde jsem žádné informace o tomto projektu nenašel. Tudíž jsem se rozhodl, že se již nadále nebudu trápit nad řešením, proč mi daný klient nefunguje, a dal jsem přednost následujícímu klientu, zvaného Jappix-Mini. Ten

[21] AGPL (Affero General Public Licence) – Licence pro svobodný software.
[22] Apache Server – softwarový webový server s otevřeným kódem.

byl už o něco jednodušší na implementování. I když nastalo pár situací, kdy jsem musel sáhnout po alternativních řešeních. Nakonec jsem zdárně vše naimplementoval a spustil. Celou implementaci i případné doplňující skripty popisuji v kapitole 5.

5 Implementace webového klienta a podpůrných skriptů

Celý XMPP klient Jappix se skládá z několika částí klientského rozhraní. První částí je webové rozhraní určené pro mobilní telefony a zařízení, nazývaný Jappix-Mobile. Druhou hlavní částí je webový klient, který obsahuje mnoho užitečných pomůcek a rozšíření, jehož grafické rozhraní je řešeno přes jedno hlavní okno webového prohlížeče. Pro naši práci jsem si vybral jeho třetí rozšíření, které je označováno jako Jappix-Mini [17]. Jak už jsem zmiňoval v minulé kapitole, jeho klientské rozhraní je umístěno ve spodní části webového prohlížeče a nikterak nezasahuje do zbytku obsahu stránky (Obrázek 4.2). Všechny tyto části mají společné jádro aplikace, a tudíž je tento klient distribuován v jednom instalačním balíčku. Při naimplementování tohoto klienta budeme mít možnost využít jakoukoliv jeho část. Ovšem my se budeme soustředit jen na implementaci Jappix-Mini rozšíření do webového uživatelského rozhraní Virtlabu.

5.1 Instalace Jappix XMPP klienta

Jappix klient budeme stejně jako Openfire server instalovat vždy na server dané lokality, tedy v našem případě virtuální lokality Most. Instalace Jappix klienta je rozdělena celkem do pěti bodů. První tři části jsou instalace prvků, které jsou potřebné pro správný chod klienta a celé komunikační XMPP sítě vůbec. Jedná se o nastavení a instalaci webového serveru Apache, XMPP serveru a BOSH serveru. Další část zahrnuje instalaci a nastavení Jappix klienta, a poslední část se věnuje implementaci rozhraní Jappi-Mini klienta, do jiných internetových stránek. Všechny tyto části si následně popíšeme a naimplementujeme do našeho systému.

5.1.1 HTTP Server Apache

Asi nejdůležitějším prvkem pro chod webového klienta a celého webu jako celku vůbec, je mít nainstalovaný webový server. Webový server se stará o příjem, zpracování a odesílání HTTP požadavků. Ve zkratce, klient (např. vzdálený počítač) zasílá požadavek na webový server. Ten má na svém pevném disku uložené jednotlivé webové stránky, webový server požadavek přijme, zpracuje a vrátí požadovaná data, či informace, které jsou na disku uloženy. Bez takového serveru by se žádné webové stránky neobešli.

Pro naše potřeby jsem si vybral webový server Apache a to z jednoduchého důvodu. Již je na každé lokalitě nainstalován a používán pro webové rozhraní Virtlabu. Proto v tomto bodě celá instalace odpadá a my jej máme připravený pro naše potřeby.

5.1.2 XMPP Server Openfire

Dalším nutným prvkem, který je důležitý pro chod, nebo spíš pro následnou komunikaci mezi uživateli, je XMPP server. Jak jsem již zmiňoval, XMPP protokol je založen na architektuře klient-server, kdy komunikace mezi klienty je vedena přes tento server. My již máme naimplementován Openfire XMPP server. Následně se budeme pomocí Jappix klienta připojovat a komunikovat s tímto serverem.

Při prohlížení různých uživatelských diskuzí a dokumentací jsem narazil na Jappix-plugin pro Openfire server. Je to přednastavený Jappix klient, který získává informace o nastavení rovnou z Openfire serveru. Když se mi nabídla příležitost implementovat Jappix klienta pouhým přenesením souboru bez dalších nutných instalací, tuto možnost jsem využil. Nakopíroval jsem daný soubor do Openfire instalačního adresáře určeného pro správu doplňků a restartoval jsem server. Při přistupování k samotnému klientu bylo vše v pořádku a vše fungovalo, ovšem když jsem se snažil daného klienta implementovat do webového rozhraní Virtlabu, nastal problém, který se mi nepodařilo vyřešit. Pravděpodobně šlo o nesprávné nastavení http-bindování externího BOSH serveru. Tak jsem se vrátil k původní instalaci Jappix klienta. Více informací o tomto Openfire-pluginu naleznete na stránkách projektu [11].

5.1.3 BOSH Server Punjab

BOSH (Bidirectional-streams Over Synchronous HTTP), je technologie založena na obousměrné komunikaci přes HTTP protokol. BOSH napodobuje mnoho transportních protokolů, jako je TCP[23]. Je určený pro aplikace, které vyžadují „Push“ a „pull“ komunikaci, zpravidla jsou to weboví klienti. Je výkonnější a citlivěji reaguje oproti ostatním obousměrně založeným HTTP protokolám. BOSH byl původně vyvinut v Jabber/XMPP komunitě. Více o této technologii XEP-0124 na [12].

Openfire používá vlastní interní BOSH server. Bohužel pro použití Jappix-Mini klienta je tento BOSH server nepoužitelný. Je známo, že Jappix-Mini nebude za použití tohoto vnitřního serveru správně fungovat. Tak nám nezbývá než nainstalovat některý z externích BOSH serverů. Já jsem se rozhodl pro nainstalování BOSH serveru Punjab. [16]

Instalace tohoto serveru byla poněkud náročnější, protože ke své správné činnosti potřeboval hned několik potřebných modulů:

[23] TCP

- **Python 2.5 nebo vyšší** – Tento server je napsaný v jazyce Python[24], proto je hlavní součástí pro jeho chod podpora tohoto jazyka. Naštěstí náš operační systém už tento jazyk podporuje.
- **Twisted 2.5 nebo vyšší –** Je událostmi řízený síťový engine určený pro jazyk Python [13]. Tento engine však potřebuje ke svému chodu nainstalovat balíček zope.interface.3.8.0 [15].
- **pyOpenSSL –** je Python knihovna pro zabezpečenou komunikaci. Je to tenký obal OpenSSL[25] knihovny. Nedělá nic jiného, než že volá funkce OpenSSL knihovny. [14]

Po nainstalování těchto komponent jsem stáhl balíček a rozbalil. Následovala instalace s adresáře `/punjab` kde jsem si server rozbalil.

```
#python setup.py install
```

Poté si musíme vytvořit TAC soubor, který pojmenujeme punjab.tac. Následuje obsah souboru, kterým jsme nastavili vlastnosti BOSH serveru:

```
# ----------------------------------------------------------------
# punjab tac file
from twisted.web import server, resource, static
from twisted.application import service, internet

from punjab.httpb  import Httpb, HttpbService

root = static.File("/var/www/most.dvirtlab.net")

b = resource.IResource(HttpbService(1))

root.putChild('http-bind', b) # url for BOSH

site  = server.Site(root)

application = service.Application("punjab")
internet.TCPServer(5280, site).setServiceParent(application)
#----------------------------------------------------------------
```

Tento řádek nastavuje statický html adresář:

```
root = static.File("/var/www/most.dvirtlab.net")
```

Tento řádek nastaví url pro BOSH:

```
root.putChild('http-bind', b)
```

[24] Python
[25]

Následně připravený BOSH server spustíme příkazem:

```
#twistd -y punjab.tac
```

Nyní nám jede BOSH server a naslouchá na portu 5280. Pokud chceme použít Jappix, přes zabezpečení v podobě firewall, je třeba nastavit na webovém serveru Apache proxy. To provedeme vytvořením souboru `http_proxy.conf` pokud ho ještě nemáme vytvořený, nebo jeho upravením. Ten najdeme v adresářovém prostoru webového serveru Apache `/etc/apache2/mods-available`. Soubor bude vypadat takto:

```
<IfModule proxy_http_module>
    ProxyRequests Off
    ProxyPass /http-bind http://localhost:5280/http-bind
    ProxyPassReverse /http-bind http://localhost:5280/http-bind
</IfModule>
```

Druhý řádek `ProxyRequest Off` říká, že nezakazuje používat ProxyPass směrnice. Třetí řádek nám udává, kam se budou odrážet zaslané žádosti. Například požadavek http://localhost/http-bind/example bude vnitřně převeden do proxy žádosti http://localhost:5280/http-bind/example. Čtvrtý řádek zajistí změnu HTTP přesměrování, obdobně jako v třetím řádku.

Dále zbývá povolit modul proxy_http. To uděláme následujícím příkazem a restartujeme Apache server:

```
a2enmod proxy_http
/etc/init.d/apache2 restart
```

Tím by mělo být vše kolem BOSH serveru nastaveno. Je zde však možnost, že nastavení není naprosto funkční, proto je někdy zapotřebí provést intuitivní změny.

5.1.4 Instalace Jappix Klienta

Tři základní prvky nutné k chodu Jappix webového klienta máme nastavené. Teď už následuje samotná implementace. Rozbalíme klienta do složky, kde máme web Virtlabu `/var/www/most.dvirtlab.net`. Prvotní konfiguraci Jappix klienta provedeme přes internetový prohlížeč na adrese domény naší virtuální lokality plus název složky Jappix klienta (https://most.dvirtlab.net/jappix/). Přivítá nás instalační průvodce, pomocí kterého nastavíme Jappix klienta pro naše účely (ukázka instalačního průvodce Obázek 5.1). Průvodce má podporu českého jazyka a je vytvořen tak, aby nastavení zvládl i méně pokročilý uživatel.

První věcí o co nás instalační průvodce požádá než bude pokračovat dále je, abychom mu dovolili zapisovat data do složky `./store`, kde ukládá následnou konfiguraci a další data. To provedeme v příkazovém řádku příkazem:

```
#chmod -R 777 /var/www/most.dvirtlab.net/jappix/store
```

Druhým krokem je založení účtu administrátora pro spravování tohoto nastavení a pro případné pozdější změny.

Dále následuje hlavní nastavení Jappix klienta. Vyplníme název a popis služby, pro nás nepříliš důležitá věc, protože v Jappix-Mini klientovy se tyto údaje nezobrazují. V kolonce „nastavení zdroje“ udáváme název, z jakého klienta se přihlašujeme (user@domain.com/Zdroj). Tuto hodnotu můžeme ponechat na defaultním nastavení (Jappix). Další nastavení připojení:

- **Uzamknout hostitele** – Tím vybereme, zda chceme povolit uživatelům přihlásit se XMPP serveru. Tato nabídka se používá, pokud máme omezený BOSH server (interní BOSH server Openfire). Tohle se nás netýká tudíž odškrtneme.
- **Anonymní mód** – Povolit anonymní režim přihlášení? Nechceme, na náš Openfire server se budou moct přihlásit jen registrovaní uživatelé také odškrtneme.
- **Registrace povolena** - Nastavení zda chceme povolit registraci pomocí Jappix klienta. Tohle nastavení není pro nás důležité. Ponechali jsme zaškrtnuté.
- **Použít proxy** – Vybereme, zda chceme použít vestavěný PHP Bosh proxy server. Nechceme opět odškrtneme.
- **Šifrování** – Tady sdělíme klientovy, zda náš XMPP server podporuje HTTPS protokol, tedy šifrované spojení. Podporuje takže zaškrtneme.

Další nastavení v tomto formuláři nás nezajímají, týkají se nastavení úložiště souborů a my tuto funkci nebudeme využívat. Přejdeme na další formulář, kde nastavíme připojení k našemu Openfire XMPP serveru.

- **Hlavní hostitel** – Zde uvedeme adresu XMPP serveru, na který se chceme připojovat. My se budeme chtít připojit na náš Openfire server který má doménové jméno `most.dvirtlab.net`.
- **Hostitel skupinových rozhovorů** – Nastavení adresy skupinového chatu XMPP serveru Openfire, u nás to bude `conference.most.dvirtlab.net`.
- **Anonymní host** – Nastavení adresy pro přihlášení anonymního uživatele k XMPP serveru. Tuhle možnost máme zakázanou, ale je dobré ji napsat pro případné budoucí použití. U nás to bude `anonymous.most.dvirtlab.net`.

- **Directory host** – Zde bude adresa VJUD adresáře XMPP serveru. Tuhle službu náš Openfire server nepodporuje. Necháme nastavenou defaultní hodnotu.
- **BOSH host** – Nastavení adresy BOSH serveru, který používáme. My jsme si nastavili svůj vlastní BOSH server na který přistupujeme adresou http://most.dvirtlab.net:5280/http-bind.

Poslední formulář obsahuje informace o potřebných modulech, které jsou potřeba pro rozšíření funkcí Jappix. Jappix klient bude pracovat i bez těchto dalších modulů, ale některé funkce budou nedostupné. Jedná se o modul GD libary a cURL libary. Po dokončení celého nastavení, Jappix vygeneruje cache soubory, což může chvíli trvat. Poté jsme přesměrováni na hlavní stránku Jappix klienta. Zde se můžeme přihlásit na svůj XMPP účet, který máme zřízený na Openfire serveru. Tím vyzkoušíme, zda byla konfigurace úspěšná, nebo zda je třeba provést nastavení znova. Celé nastavení máme uložené v XML souborech ve složce `./store/conf` Jappix klienta.

Obrázek 5.1 Instalační průvodce Jappix XMPP klienta

5.1.5 Implementace Jappix-Mini klienta do webového rozhraní

Nyní máme vše připraveno pro implementaci Jappix-Mini klienta do webového rozhraní Virtlabu. Importace Jappix-Mini rozhraní do jakékoliv webové stránky se provádí následujícím kódem, který musí být vložen do hlavičkové části HTML kódu:

```
<script type="text/javascript" src="jappix/js/jquery.js"></script>
<skript type="text/javascript"
src="jappix/php/get.php?l=en&amp;t=js&amp;g=mini.xml"></script>
```

Tyto dva řádky nám umožňují přístup do javascriptového jádra Jappix aplikace a tím ovladání Jappix-Mini rozhraní klienta. K Přihlášení do Jappix-Mini použijeme tento javascriptový kód:

```
launchMini(true, false, window.location.hostname, „jid“, „passw“);
```

Prvním atributem říkáme, zda se chceme automaticky přihlásit. Druhý zda chceme po přihlášení rozvinout Roster uživatele. Třetí atribut nám vrací doménové jméno aktuální adresy, které dále používáme jako doménu pro připojení k XMPP serveru. Čtvrtý a pátý atributem zadáváme přihlašovací jméno a heslo k XMPP účtu. My budeme tento kód ve webovém prostředí Virtlabu používat obdobně, což si popíšeme v další podkapitole.

5.2 Implementace do webového rozhraní Virtlabu

Celé webové rozhraní systému Virtlab je napsané ve skriptovacím jazyku PHP. Abych mohl do tohoto webového rozhraní nějak zasáhnout svými skripty pro ovládání XMPP klienta, musel jsem pochopit celou syntaxi a smysl rozvržení těchto stránek. Musím poděkovat a ocenit práci Ing. Jana Vavříčka, který tyto stránky napsal. Pro mě jako pro úplného začátečníka v tomhle oboru psaní webových stránek, jsem strukturu stránek rychle pochopil, a tudíž jsem mohl snadno dopsat a doplnit příslušné moduly rozhraní pro XMPP klienta.

Před samotným vývojem a implementací, jsem si musel stanovit, co vlastně chci, aby mé rozšíření webového rozhraní umělo. Potřeboval jsem přidělat tyto komponenty:

- **Rozhraní** – Vytvoření rozhraní pro prvky přihlášení, informace, seznam místností a rozšíření.
- **Registrace** – Registrace uživatele na XMPP Openfire server při vytváření nového uživatele Virtlabu. Tohle má na starosti administrátor Virtlabu.

- **Přihlášení/odhlášení** – Prvek řešící přihlášení a odhlášení uživatele ke klientu Jappix-Mini.
- **Informace** – Informační stránka o XMPP protokolu a jeho chodu ve webovém rozhraní Virtlab.
- **Seznam Místností –** Prvek zobrazující seznam víceuživatelských místností pro daného uživatele.
- **Rozšíření –** Administrační rozhraní zabývající se hromadnou registrací uživatelů.

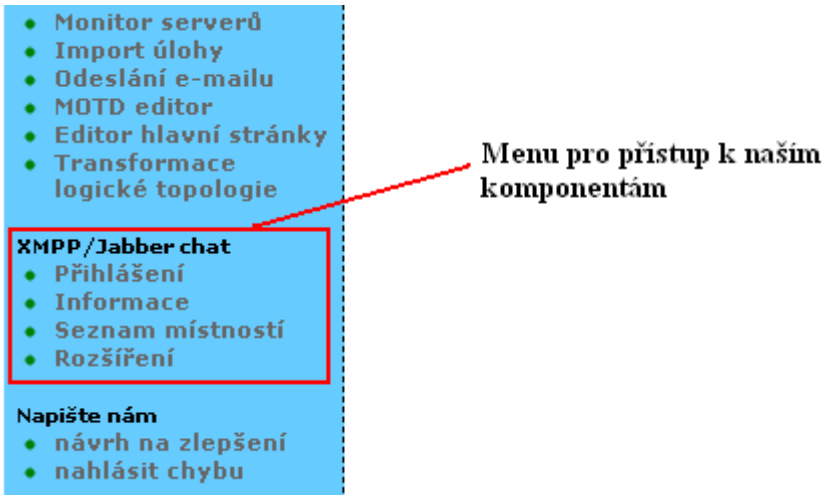
Soubory, které ovládají celé webové rozhraní Virtlabu, se nachází v základním adresáři `/var/www/most.divrtlab.net`. Tady budeme upravovat dané soubory. Všechny tyto upravované soubory budou přidány do přílohy na CD/DVD i s danou adresářovou architekturou.

5.2.1 Webové rozhraní

Webové rozhraní systému Virtlab se skládá ze 4 prvků. Hlavičky, ve kterých se definují různé komponenty pro správný chod celého webového rozhraní. Menu, které je umístěno na levé straně a umožňuje nám orientaci v celém rozhraní. Hlavního těla stránky, kde se zobrazují potřebné uživatelské informace a komponenty pro ovládání systému Virtlab. Poslední prvkem je tzv. tail , který není prozatím ničím využitý. Kompletní webové rozhraní Virtlabu řídí soubor `index.php`, který svými funkcemi načítá a sestavuje výchozí rozhraní. Do tohoto souboru jsem připsal kód pro načítání komponent těla stránky. Jedná se o načtení komponent stránek přihlášení, informace, seznam místností a rozšíření.

Načítání jednotlivých komponent stránek je indexovaný pomocí čísel, které se přiřadí jednotlivé komponentě. Toto indexování řeší soubor `/class/virtlabWeb.php.inc`, do kterého jsem připsal kód, definují index dané komponenty.

Pro zahrnutí našich definovaných komponent do tohoto rozhraní jsem dané prvky upravil, tak abych nijak neovlivnil chod stránky a také abych zachoval danou syntaxi a strukturu. Do menu jsem zakomponoval část pro přístup k našim definovaným komponentám (Obrázek 5.2). Úprava se týkala souboru `/virtlab/menu.php`.



Obrázek 5.2Menu pro komponenty XMPP komunikační sítě

V hlavičce jsem připsal část kódu, která nám řeší importaci našeho webového Jappix-Mini klienta, kterého budeme ve webovém rozhraní Virtlabu používat pro komunikaci mezi uživateli. Jedná se o tento kód:

```
<script type="text/javascript" src="jappix/js/jquery.js"></script>
<skript type="text/javascript"
src="jappix/php/get.php?l=en&amp;t=js&amp;g=mini.xml"></script>
```

Úprava se týkala souboru `/include/pages/head.php.inc`.

Webové rozhraní má podporu vícejazyčnosti (Čeština, Angličtina). Tuto funkci řeší soubor `/class/virtlabLanguage.php.inc`, ve kterém jsem doplnil českou a anglickou verzi pro užití textových prvků do mých vytvořených komponent.

Tyto všechny úpravy a nastavení jsem moc nerozepisoval, protože podle mě nemají nějakou větší zásluhu na chod komponent pro řízení Jappix-Mini klienta. Tyto úpravy vytvářejí rozhraní a rozšíření pro tyto komponenty. Důležité skripty budou obsahovat těla těchto komponent, které následně popíši.

5.2.2 Registrace uživatelů

V souboru `/virtlab/users-new.php` jsem dopsal PHP skript, který řeší registraci uživatelů na XMPP Openfire server. Při vytváření nového uživatele do systému Virtlab se pomocí tohoto skriptu zároveň registruje uživatel na XMPP Openfire server. Protože při vytváření nového uživatele do systému Virtlab je možné ověřovat identitu pomocí LDAP (a to u XMPP serveru nelze, protože to nemáme nastavené), nastavil jsem generování hesla a zaslání registračních údajů pro XMPP účet na e-mail uživatele. Ten si poté může heslo změnit v hlavním Jappix klientu.

5.2.3 Přihlášení

Přihlášení ke XMPP účtu nebude probíhat automaticky po přihlášení uživatele do webového rozhraní systému Virtlab, ale daný uživatel bude mít možnost přihlásit se ke svému XMPP účtu pomocí vytvořeného formuláře (Obrázek 5.3). Po přihlášení se vytvoří PHP proměnné typu `$_SESSION['JID-xmpp']`, `$_SESSION['PASSW-xmpp']`, do kterých se uloží přihlašovací jméno a heslo uživatele. Dále se vytvoří proměnná `$_SESSION['logged-xmpp']`, do které budeme ukládat hodnotu ‚1' pokud je daný uživatel přihlášen.

Obrázek 5.3 Formulář pro přihlášení k XMPP účtu

Do souboru `/include/pages/tail.php.inc` jsem přidal skript, pomocí kterého se automaticky přihlašujeme k Jappix-Mini klientu. Tento skript se provede vždy při načtení nové stránky, tím docílíme, že jsme neustále přihlášení do klienta, i když přecházíme ze stránky na stránku. Tento skript je aktivní jen pokud se uživatel správně přihlásil. To nám udává naše session proměnná `$_SESSION['logged-xmpp']`.

Ovšem musíme vyřešit jeden problém s přihlášením. Protože klient bude používat více uživatelů, musíme funkci napsat s použitím proměnných, které se budou měnit podle daného uživatele. Takto vypadá funkce pro přihlášení do klienta:

```
launchMini(true, false, window.location.hostname, jid, passw);
```

Poslední dvě proměnné nám udávají JID a heslo uživatele, který se chce přihlásit k Jappix-Mini klientu. Tady využijeme naše vytvořené session proměnné `$_SESSION['JID-xmpp']`, `$_SESSION['PASSW-xmpp']`, které místo nich použijeme. Tímto docílíme přihlašování pro daného uživatele, který se právě k tomuto klientu přihlásil.

Celé přihlášení k Jappix-Mini klientu je řešeno dvěma soubory. První soubor `/include/javascript/xmpp_login.js` obsahuje Javascriptové funkce pomocí nich se přihlásíme či odhlásíme z Jappix-Mini klienta. Druhý soubor `/virtlab/xmpp_login.php`

obsahuje kód pro zobrazení formuláře přihlášení. Nepovedené přihlášení poznáme podle zobrazení ikony vykřičníku a hlášky Error tam kde je zobrazený Jappix-Mini klient.

5.2.4 Informace

Tato komponenta slouží pro zobrazení základních informací o používání Jappix-Mini klienta a případných komponent. Informace se zobrazují pomocí PHP kódu, který je načítán ze souboru `/virtlab/xmpp_info.php`.

5.2.5 Seznam víceuživatelských místností

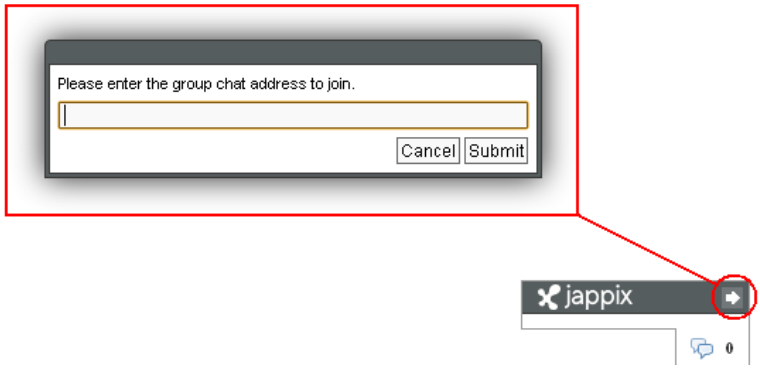
Tato komponenta se měla starat o vypsání všech víceuživatelských místností ze všech lokalit dostupných pro daného uživatele. Bohužel tuto komponentu jsem nezrealizoval, protože jsem nenašel dané funkce v Jappix architektuře, které by to umožnili.

Má představa této komponenty byla následující. Měl se zobrazit seznam MUC místnosti, který obsahoval statickou místnost dané lokality (každá lokalita má jednu statickou místnost kde má přístup každý uživatel) a seznam rezervovaných úloh na kterých se daný uživatel podílí. Každý záznam seznamu měl mít tlačítko „vstoupit“. Tohle tlačítko by vyvolalo danou funkci pro přihlášení do víceuživatelské místnosti s daným jménem záznamu.

Každá rezervovaná úloha má své jedinečné jméno. Tohle jméno by bylo použito jako jméno MUC místnosti. Smysl by byl v tom, že uživatelé, kteří se podílejí na rezervované úloze, by se mohli přihlásit do této místnosti a diskutovat spolu o řešení praktické úlohy. Přihlášením do MUC místnosti, která neexistuje, se vytvoří místnost s daným jménem a uživatel je do ní následně přihlášen. Ovšem každá takhle vytvořená místnost žádá potvrzení o nastavení charakteru místnosti, dokud není místnosti potvrzeno nastavení, uživatelé v ní nemohou komunikovat. Tato možnost potvrzení nastavení MUC místnosti v rozhraní Jappix-Mini chybí, a tudíž tuhle variantu nemůžeme využít.

Můžeme využít alternativní metodu, kdy se přihlásíme do hlavního prostředí Jappix klienta, tam už je tahle možnost vytváření víceuživatelských místností podporována.

Pro každou lokalitu už je staticky při instalaci Openfire XMPP serveru vytvořena MUC místnost, do které má přístup každý uživatel dané lokality ve které je zaregistrován. Přihlášení do skupinového chatu provádíme kliknutím na šipku v Rosteru Jappix-Mini klienta, kdy nám vyskočí formulář pro vyplnění jména místnosti, kam se chceme přihlásit (Obrázek 5.4).



Obrázek 5.4 Přihlášení do skupinového chatu

5.2.6 Administrační rozšíření

Tato komponenta má být určena pro administrátory systému Virtlab. Má usnadnit prvotní registraci všech uživatelů systému Virtlab dané lokality na XMPP Openfire server. Komponenta není zatím realizována, je to jen návrh na další rozšíření pro ulehčení registrací uživatelů.

Má představa o provedení hromadné registrace na XMPP server. Vytvoříme si seznam všech uživatelů systému Virtlab pro danou lokalitu. Poté postupně procházíme jednotlivě uživatele, a dotazujeme se XMPP serveru zda uživatel již existuje. Pokud účet s daným jménem na serveru neexistuje, zavolá se funkce pro registraci uživatele na Openfire XMPP server. Toto rozšíření dále může kontrolovat propojenost uživatelů mezi systémem Virtlab a Openfire serverem.

6 Závěr

V této práci jsem se zabýval výběrem XMPP serveru a následně webového klienta, pomocí nichž jsem vytvořil komunikační síť pro uživatele systému Virtlab. Tuto komunikační síť se mi podařilo do systému Virtlab úspěšně implementovat. Při volbě webového XMPP klienta jsem byl nucen použít služeb Jappix-Mini, jelikož byl svými vlastnostmi nejblíže mým požadavkům, respektive požadavkům na implementaci do webového rozhraní Virtlabu. O těchto požadavcích, které jsem si stanovil, se zmiňuji v kapitole 4. Jappix-Mini však nebyl naprosto ideálním řešením, poněvadž nedisponoval funkcí pro zobrazení seznamu dostupných víceuživatelských místností.

Během vypracovávání této bakalářské práce jsem se potýkal s různými problémy. Jedním z hlavních problémů byla neschopnost správného komunikování Jappix-Mini klienta s interním BOSH serverem v Openfire. Proto jsem byl nucen nainstalovat a uvést do provozu externí BOSH server Punjab, což je názorně popsáno v podkapitole 5.1.3.

Celá implementace a následné testování probíhalo na virtuální lokalitě Virtlabu, která je určena především pro vývoj, před nasazením do reálného prostředí.

Z důvodu neexistence zcela ideálního webového klienta by bylo vhodné navázat na tuto práci vytvořením vlastního webového klienta, který bude disponovat všemi potřebnými funkcemi, jako je například již zmíněný seznam víceuživatelských místností a mnoho dalších.

Použitá literatura

[1] *Virtuální laboratoř počítačových sítí - VirtlabWiki* [online]. [cit. 2012-05-03]. Dostupné z: http://infra2.cs.vsb.cz/vl-wiki/index.php/Virtuální_laboratoř_počítačových_sítí

[2] *Xmpp - Jabber.cz Wiki* [online]. [cit. 2012-05-03]. Dostupné z: http://www.jabber.cz/wiki/Xmpp

[3] *The XMPP Standards Foundation* [online]. [cit. 2012-05-03]. Dostupné z: http://xmpp.org/

[4] Extensible Messaging and Presence Protocol. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-05-03]. Dostupné z: http://en.wikipedia.org/wiki/Xmpp

[5] *Ejabberd - High Performance Instant Messaging - ProcessOne* [online]. [cit. 2012-05-03]. Dostupné z: http://www.process-one.net/en/ejabberd/

[6] *Jabberd14 XMPP/Jabber server daemon* [online]. [cit. 2012-05-04]. Dostupné z: http://www.jabberd.org/

[7] *Prosody IM* [online]. [cit. 2012-05-03]. Dostupné z: http://prosody.im/

[8] *Ignite Realtime: Openfire Server* [online]. [cit. 2012-05-03]. Dostupné z: http://www.igniterealtime.org/projects/openfire/

[9] *Ijab - An ajax web jabber client. Jabber Web Chat. WebIM.* [online]. [cit. 2012-05-03]. Dostupné z: http://code.google.com/p/ijab/

[10] *Jappix social cloud* [online]. [cit. 2012-05-03]. Dostupné z: http://jappix.org/

[11] *Ignite Realtime: Jappix for Openfire plugin* [online]. [cit. 2012-05-03]. Dostupné z: http://community.igniterealtime.org/docs/DOC-2195

[12] *XEP-0124: Bidirectional-streams Over Synchronous HTTP (BOSH)* [online]. [cit. 2012-05-03]. Dostupné z: http://xmpp.org/extensions/xep-0124.html

[13] *Twisted* [online]. [cit. 2012-05-03]. Dostupné z: http://twistedmatrix.com/trac/

[14] *PyOpenSSL in Launchpad* [online]. [cit. 2012-05-03]. Dostupné z: https://launchpad.net/pyopenssl

[15] *Index of Packages: Python Package Index* [online]. [cit. 2012-05-03]. Dostupné z: http://pypi.python.org/pypi/zope.interface

[16] *Twonds/punjab · GitHub* [online]. [cit. 2012-05-03]. Dostupné z: https://github.com/twonds/punjab

[17] *Try - Jappix Mini, a great mini-chat for your website* [online]. [cit. 2012-05-03]. Dostupné z: https://mini.jappix.com/

Přílohy na CD-ROM

1) Text této bakalářské práce v elektronické podobě.
2) Zdrojové kódy řešené problematiky